

Semiannual Technical Summary

AD-A147 638

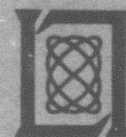
Distributed Sensor Networks

31 March 1984

Lincoln Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

LEXINGTON, MASSACHUSETTS



Prepared for the Defense Advanced Research Projects Agency
under Electronic Systems Division Contract F19628-80-C-0002.

Approved for public release; distribution unlimited.

DTIC
ELECTE
NOV 23 1984
S E D

DTIC FILE COPY

84 11 19 101

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract F19628-80-C-0002. (ARPA Order 3345).

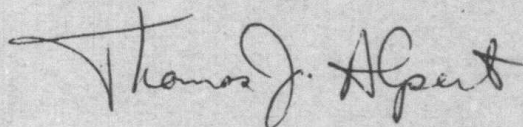
This report may be reproduced to satisfy needs of U.S. Government agencies.

The views and conclusions contained in this document are those of the contractor and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the United States Government.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

A handwritten signature in dark ink, reading "Thomas J. Alpert". The signature is fluid and cursive, with the first name "Thomas" and last name "Alpert" clearly legible.

Thomas J. Alpert, Major, USAF
Chief, ESD Lincoln Laboratory Project Office

Non-Lincoln Recipients

PLEASE DO NOT RETURN

Permission is given to destroy this document
when it is no longer needed.

**MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY**

DISTRIBUTED SENSOR NETWORKS

**SEMIANNUAL TECHNICAL SUMMARY REPORT
TO THE
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY**

1 OCTOBER 1983 — 31 MARCH 1984

ISSUED 3 OCTOBER 1984

Approved for public release; distribution unlimited.

LEXINGTON

MASSACHUSETTS

ABSTRACT

This report describes the work performed on the DARPA Distributed Sensor Networks Program at Lincoln Laboratory during the period 1 October 1983 through 31 March 1984.

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



TABLE OF CONTENTS

Abstract	iii
List of Illustrations	vii
I. INTRODUCTION AND SUMMARY	1
II. WIDEBAND ARRAY PROCESSING FOR DIRECTION DETERMINATION,	3
A. Algorithm Evaluation	3
B. Real-Time Test-Bed Implementation	5
III. TV SENSOR SUBSYSTEM,	9
IV. DISTRIBUTED TRACKING,	11
A. Experiments	11
B. Distributed Multisite Tracking Algorithms	13
C. Test-Bed Implementation of New Distributed Algorithms	13
V. KNOWLEDGE-BASED DATA INTERPRETATION,	17
VI. COMMUNICATIONS,	19
A. Radio Communication	19
B. Local Area Network	22
VII. SELF-LOCATION,	23

LIST OF ILLUSTRATIONS

Figure No.		Page
II-1	Azimuth Measurements as a Function of Time for Two-Helicopter Experiment. (a) Scenario. (b) New Wideband Algorithms. (c) Old Frequency-Wavenumber Algorithms.	4
II-2	Azimuth Measurements from Wideband Algorithm for the Flyby of a Fixed-Wing Jet Aircraft	6
II-3	Power as a Function of Azimuth as Produced by Wideband Algorithm Applied to Jet Data with Large Signal-to-Noise Ratio	6
II-4	Observation Intervals for Parametric Study of Wideband Algorithm	7
IV-1	Software Organization and Data Flows for Initial Two-Node Acoustic Tracking Algorithms	12
IV-2	Software Organization and Data Flows for New Distributed Tracking System	14
V-1	Organization of Knowledge-Based DSN Data Interpretation in Terms of the Expectation-Formation and Diagnosis Function	18
VI-1	Organization of Software to Implement DSN Radio Broadcast Protocol	20
VI-2	Two-Node Radio Communication Hardware/Software Test Set-Up	21
VII-1	Typical Two-Dimensional Network Layout with Contour N: Showing Extent of Communication Region for Node i	24

DISTRIBUTED SENSOR NETWORKS

I. INTRODUCTION AND SUMMARY

The Distributed Sensor Networks (DSN) program is aimed at developing and extending target surveillance and tracking technology in systems that employ multiple spatially distributed sensors and processing resources. Such a system would be made up of sensors, data bases, and processors distributed throughout an area and interconnected by an appropriate digital data communication system. The detection, tracking and classification of low flying aircraft has been selected to develop and evaluate DSN concepts in the light of a specific system problem. A DSN test bed has been developed and is being used to test and demonstrate DSN techniques and technology. The overall concept calls for a mix of sensor types. The initial test bed sensors are small arrays of microphones at each node augmented by TV sensors at some nodes. This Semiannual Technical Summary (SATS) reports results for the period 1 October 1983 through 31 March 1984.

During this reporting period, we completed experiments to validate our new wideband acoustic direction finding algorithm and have begun to implement the algorithm for real-time use in the test-bed nodes. The experiments verified that the algorithm performance was significantly superior to that of our previous frequency-wavenumber algorithm for single and multitarget scenarios and that it performed in a satisfactory manner for broadband jet sources. This was accomplished using recorded acoustic data for three different scenarios. These were (1) a single helicopter flyby, (2) a two-helicopter flyby and (3) a single fixed-wing jet aircraft flyby. Real-time software to implement the algorithm has been coded and is being debugged in a non-real-time test environment. Section II reports on this work in more detail.

Section III reports progress in the development of a TV sensor subsystem for the test bed. All major hardware elements are on hand and the subsystem is under construction.

Work in the area of distributed tracking has involved test-bed experimentation with previously implemented two-node tracking algorithms, integration of tracking and track initiation algorithms of our new tracking system that is based upon modern distributed estimation concepts, and efforts directed at test-bed implementation of the new system during the next reporting period. This work is reported in Section IV. The test-bed experimentation confirmed the expected tracking performance of the algorithms, provided a test of test-bed hardware and system software, helped identify useful test-bed modifications and provided useful information to aid in the further development and implementation of distributed tracking methods using the test bed. When the implementation of the new distributed tracking algorithms is completed, it will constitute the only example of such a general distributed tracking system of which we are aware.

Progress in the area of knowledge-based DSN data interpretation is reported in Section V. The work has focused upon a meta-level interpretation system to serve as a user interface. The approach is based upon a expectation/diagnosis paradigm in which discrepancies between DSN outputs and expectations are diagnosed to provide data interpretations. Two experimental rule-based diagnosis systems have been implemented to investigate the limits of a rule-based approach, and we are designing a more advanced system that incorporates other reasoning methods as well.

Progress in the area of communications is reported in Section VI. The development of a radio capability for the test bed has continued. Software for a broadcast protocol has been designed in detail. Much of it has been coded and is now being tested. A two-node test set-up is being used to debug the hardware interface between radio units and a standard DSN node and will subsequently be used for real-time testing of the broadcast protocol. In addition, first steps have been taken to integrate an Ethernet local network capability into the test bed to enhance its experimental utility.

Section VII reports work on the development of new distributed self-location algorithms that are based upon distributed estimation theory.

II. WIDEBAND ARRAY PROCESSING FOR DIRECTION DETERMINATION

Experiments have been conducted to further evaluate the new wideband DSN signal processing algorithm that was described in the last Semiannual Technical Summary.* Based upon the results of those experiments, we have proceeded with the real-time test-bed implementation of the algorithm. In addition, a technical paper was presented at the 1984 International Conference on Acoustics, Speech and Signal Processing.†

A. ALGORITHM EVALUATION

Algorithm evaluation experiments included a comparison of the performance of the old and new algorithms for a two-helicopter experiment, a verification of satisfactory performance by the new algorithm for a relatively high-speed broadband jet source and a parametric investigation of the performance of the new algorithm using single helicopter data.

The two-helicopter experiments utilized recorded data sets that were described in the previous SATS. The general conclusion was that the new algorithm provided a greater detection range and reduced the variance of the estimated azimuths. The azimuth measurements produced by the two algorithms for one data set are shown in Figure II-1. The two-helicopter results are consistent with the superior performance we had previously found for the new algorithm when applied to single aircraft data sets.

Data from a relatively high-speed, low-flying jet aircraft have been used to confirm that the new algorithm will perform in a satisfactory manner for such a scenario. The data were obtained at Edwards Air Force Base, California, in cooperation with the DARPA/Air Force sponsored Air Vehicle Survivability Evaluation program at Lincoln Laboratory. One of the mobile DSN nodes was used as the data acquisition system.

An eight-microphone array with a six-meter aperture was used to obtain data. The aircraft flew past the array at an approximately constant velocity of 300 knots. The altitude was 200 ft above ground level and the closest point of approach was offset 300 ft from the array. Seventy seconds of recorded data were analyzed. At the beginning of the interval, the aircraft was at a range of 7 km and was inbound towards the array. At the end of the time interval, the aircraft was 3 km away from the array and outbound away from it. Algorithm parameters were set to respond to signals in the 100 to 200 Hz band, to average for 1 second for each observation and to process the data in 128 point blocks, corresponding to about 0.06 seconds.

* Semiannual Technical Summary, Distributed Sensor Networks Program, Lincoln Laboratory, M.I.T. (30 September 1983).

† S.H. Nawab, F.U. Dowlal and R.T. Lacoss, "A New Method of Wideband Sensor Array Processing," Proc. ICAASP 84, 1, 4.12.1-4.12.4, 19-21 March 1984.

SCENARIO

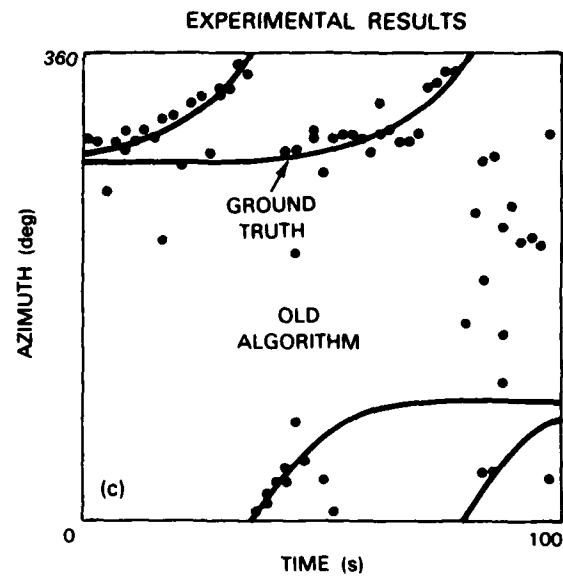
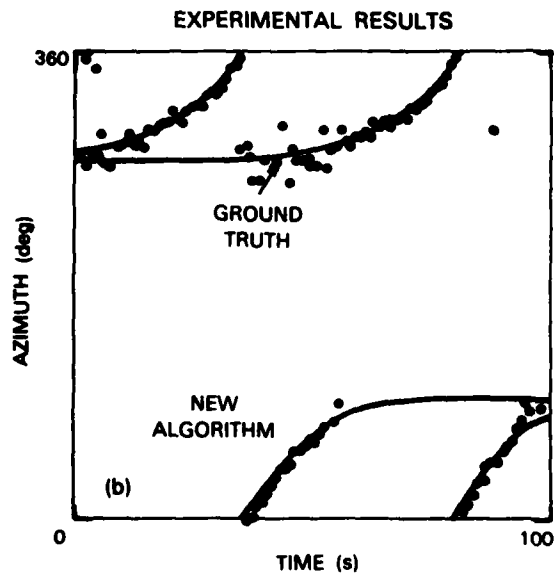
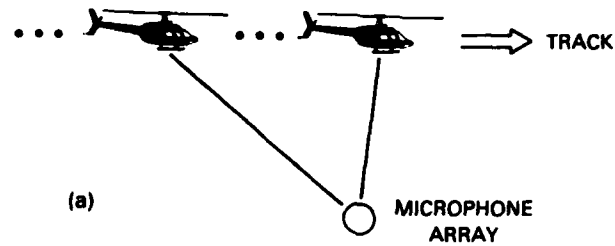


Figure II-1. Azimuth measurements as a function of time for two-helicopter experiment. (a) Scenario. (b) New wideband algorithms. (c) Old frequency-wavenumber algorithms.

137626-N-01

Figure II-2 illustrates the azimuth measurements produced by the new wideband algorithm. These measurements are in agreement with the known aircraft track, which was independently obtained from radar measurements. Figure II-3 shows the integrated wavenumber power at time = 45 seconds of Figure II-2. The figure illustrates the very large signal-to-noise ratio at that time when the aircraft was very close to the array.

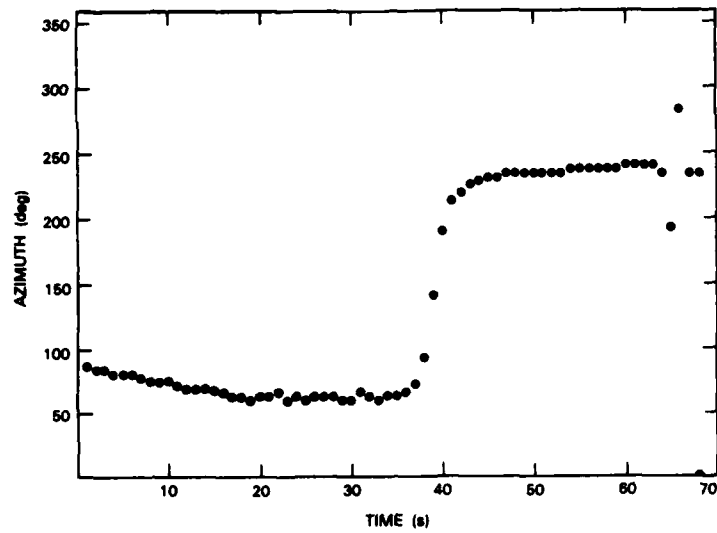
Additional experiments with a UH-1 helicopter were carried out to study the interactions between algorithm parameters and observation parameters such as signal-to-noise ratio and rate-of-change of source bearings. The algorithm parametric settings which were used were: (1) three different frequency bands (0-100 Hz, 0-200 Hz, 100-200 Hz), (2) four different block-lengths (32, 64, 128, 256 samples) and (3) four different analysis interval durations (0.5 s, 1 s, 2 s, 4 s). Four 20-second data intervals of data were used.

The data collection scenario and the UH-1 track regions corresponding to the four data intervals are shown in Figure II-4. Region 1 corresponds to data for which signal-to-noise ratio is low and source bearing is almost constant. Region 2 corresponds to data where signal-to-noise ratio is high and source bearing is almost constant. Region 3 corresponds to data where signal-to-noise ratio is high and source bearing is changing rapidly. Region 4 corresponds to data where signal-to-noise ratio is low and source bearing is changing rapidly.

For each region analyzed, we were interested in the portion of the 'parameter space' over which the algorithm was able to detect the helicopter throughout the entire 20-second interval. For region one, this required the longer analysis intervals, shorter block lengths and that the frequencies below 100 Hz be included. The longer analysis intervals were required to build up signal-to-noise ratios by time averaging. The lower frequency data are required because lower frequency data suffer less propagation attenuation. Satisfactory detection was obtained for region two for all parameter settings that were tested. For region three, any frequency band could be used, but short block lengths and analysis intervals were required due to the time-varying nature of the signals. For region four, no single parameter setting resulted in detection over the entire 20 seconds. For example, long averaging times could not be used to increase signal-to-noise ratios because of the time-varying nature of the signals. These results are self-consistent and do not represent any unexpected or unacceptable sensitivity to scenario or algorithm parameters.

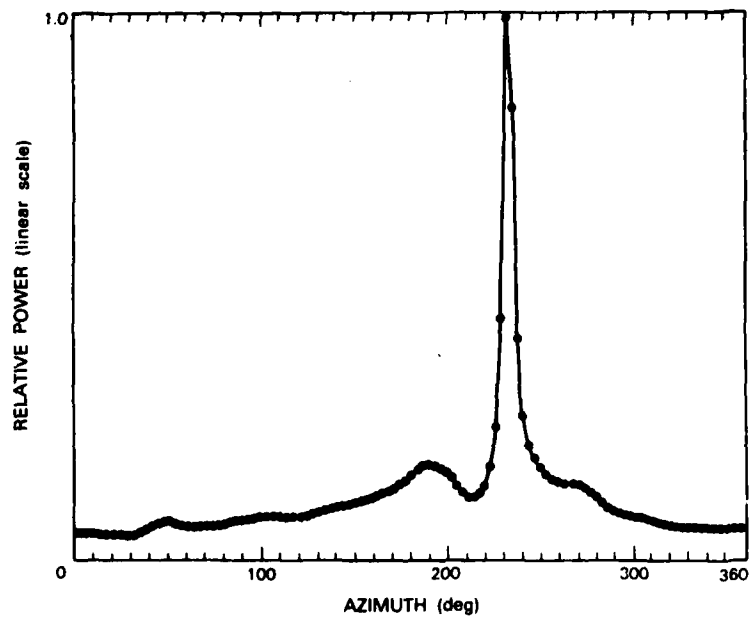
B. REAL-TIME TEST-BED IMPLEMENTATION

The implementation of real-time test-bed versions of the new wideband algorithms was started during this reporting period. This is a two-stage process starting first with a UNIX based version for our PDP11/70 system with attached FPS120B array processor and then proceeding to the real-time implementation for the nodal signal processing subsystems. The signal processing subsystems each contain a PDP11/34 and an FPS120B. Software designs and implementation plans have been developed. The UNIX based version has been implemented and is undergoing testing and refinement.



143180-N

Figure II-2. Azimuth measurements from wideband algorithm for the flyby of a fixed-wing jet aircraft.



143181-N

Figure II-3. Power as a function of azimuth as produced by wideband algorithm applied to jet data with large signal-to-noise ratio.

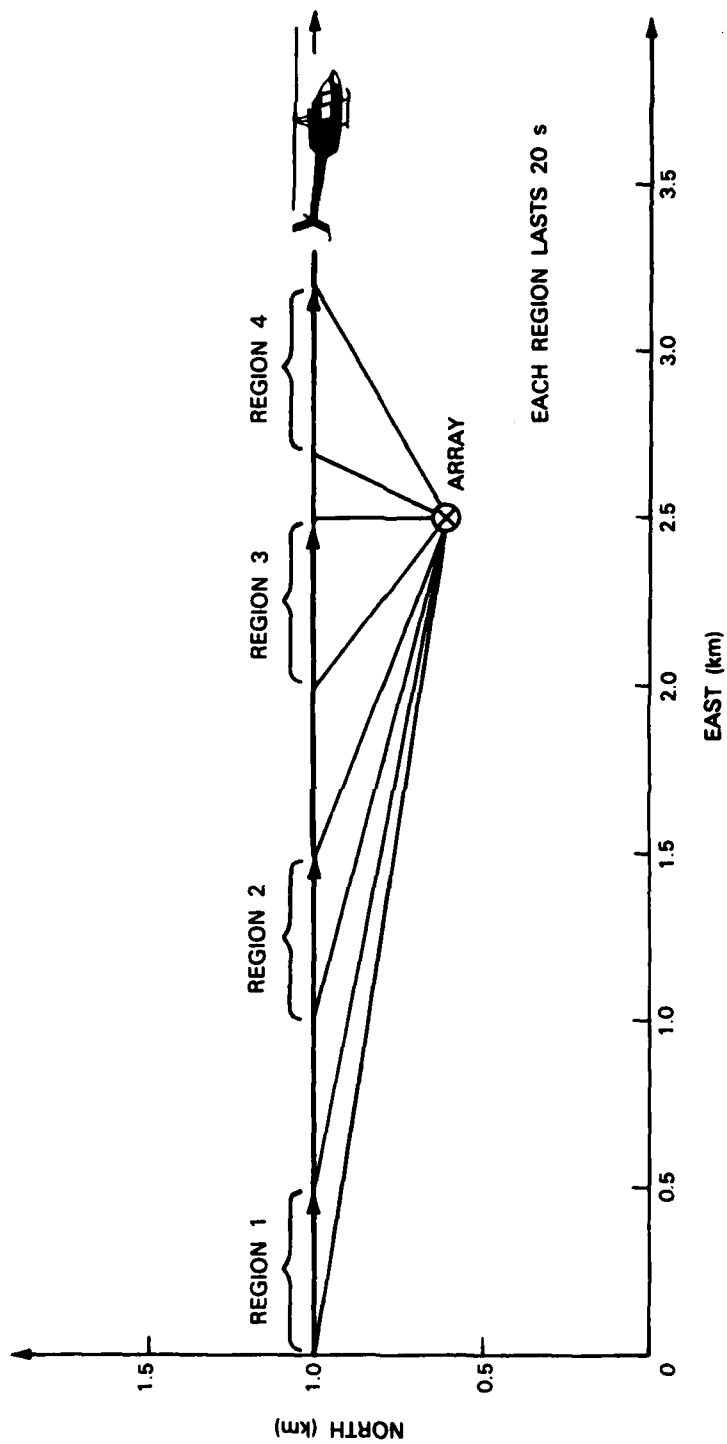


Figure II-4. Observation intervals for parametric study of wideband algorithm.

For both implementations, the bulk of the computation will be performed by the AP120B array processor. The AP120B code is identical for the PDP-11/70 versions and the PDP-11/34 test-bed node versions. Therefore, we have been able to develop the application code using our PDP-11/70 system in parallel with acquisition and installation of system software for the real-time nodes. The UNIX based PDP-11/70 system is now operational. It uses UNIX files or magnetic tapes as acoustic data sources. As a byproduct of the UNIX implementation, we gained a very fast signal processing system on the PDP-11/70 for evaluating the performance of the algorithm.

Unlike the implementation of the older signal processing algorithms, the new one includes the option to dynamically adjust algorithm parameters. The older algorithms essentially provided for parameter settings to be changed only at compile time. This is important for two reasons. First, it significantly improves the usability of the experimental test bed in terms of the ability to accommodate different parameter settings. Second, we may eventually wish to have higher level processes adjust signal processing parameters to improve system performance. In such a case, the system gains some knowledge about the environment and controls the signal processing subsystem by changing its parameters.

The UNIX version is the same as the real-time nodal version except for the control program. The nodal version will provide for data to be read from either the digital tape, the A/D system attached to microphones or from disk. It will also provide for recording of both raw and processed acoustic data.

RSX-11 was selected to support the real-time nodal version of the algorithm. This operating system provides the necessary real-time support for data collection and signal processing. A disk-based version provides a good environment for software development and testing while a diskless version is available for use in test bed nodes. Reliable vendor-supplied and -supported software is available for hardware diagnosis and for use of FPS120B array processors within the RSX-11 environment. Essential functions provided by RSX include inter- and intra-process synchronization, inter-process communications and direct hardware interrupt vectoring to user tasks. A disk has been added to the PDP-11/34 in one of the test-bed nodes and the disk based RSX-11M version of the operating system has been installed. Installation and test of the array processor support software is in progress.

III. TV SENSOR SUBSYSTEM

The assembly of hardware to make up a remote controlled closed-circuit TV camera/display and image processing subsystem is underway. This equipment will be used to conduct feasibility experiments with the DSN system to demonstrate how acoustic sensors could be used to direct visible imaging sensors to a target and how those imaging sensors could be used to enhance the overall performance of a DSN system.

All video equipment for a DSN video subsystem has either been received or is on order. Equipment on hand includes: an environment-resistant TV camera with an f/1.8 10:1 zoom lens, a pan and tilt mount with an eight-position controller, a 15-inch black and white rack-mounted TV monitor, a frame grabber and image processing subsystem and a microcomputer to act as host for the image processor and controller for the TV camera assembly. The digital electronics and processing elements of the video sensor subsystem consist of a number of commercially available multibus boards. A standard DSN nodal computer configuration is being used as the basis of the video subsystem. The chassis, backplane and rack for the video subsystem are on hand and hardware installation and test are underway. A pedestal is being built to support the pan and tilt-mounted camera for initial experimentation. Hardware changes are being designed so the camera zoom and focus can be adjusted and the camera can be pointed to any azimuth and elevation under remote computer control. The one outstanding item that has not yet been received is a video annotator to overlay time and other important information directly onto video frames.

IV. DISTRIBUTED TRACKING

During this reporting period, our distributed tracking research was comprised of three activities. First, tracking experiments were conducted with previously developed single-node azimuth tracking and two-node position tracking algorithms. These algorithms had been previously interfaced to the Nodal Real Time System (NRTS) software and been installed in the test-bed Standard Nodal Computers (SNC)¹. These experiments were to obtain performance information on the algorithms, to gain knowledge to guide the development of an improved tracking system for the test bed and to provide a thorough test of NRTS and of the SNCs. Second, development of new tracking algorithms¹ continued with the integration of track initiation and tracking algorithms and the preparation of a technical paper² dealing with distributed tracking. Third, we have specified the modules and data flow for a test-bed implementation of the newly developed distributed tracking algorithm. The modular design supports easy experimentation with algorithm components, including the addition of components to make use of azimuth measurements derived from TV sensors, and corrects a number of deficiencies of our initial two-node test-bed tracking system.

A. EXPERIMENTS

Tracking experiments were carried out with up to six test-bed nodes; ten runs included all six nodes. Previously developed simulation software³ was used to generate simulated acoustic data for the experiments. The use of simulated data allowed for repeatable experimentation with exactly known true target tracks. Tracking performance was assessed by comparing computed tracks with true tracks.

One or two helicopters were simulated for each experiment, each flying a straight course at 30 meters per second. The typical duration of an experiment was 5 minutes, allowing each helicopter to fly 9 km through a network of up to six nodes. Experiments were performed for realistic levels of missed and false detections as well as with idealized sensors that missed no detections and provided no false detections.

Serial lines were used to provide internodal communications through a PDP-11/70 that served as a message switch. User control and display programs also ran on the 11/70. Varying degrees of broadcast connectivity were simulated, although full connectivity of all six nodes was never simulated, due to limitations of the serial lines and central switch.

The tracking algorithm was implemented as two asynchronous, data-driven processes. The flow of information between and within nodes is illustrated in Figure IV-1. The heavy lines show the primary flow of measurement or tracking information and the thin lines show secondary flows required to set parameters and initiate processing.

Two major problems were encountered with this purely data-driven implementation. First, a temporary overload of a tracking process in one node could cause it to fall behind other nodes. As a result, the network could lose synchronization with many resulting difficulties. Second, a temporary underload of the tracking process in a node could cause it to

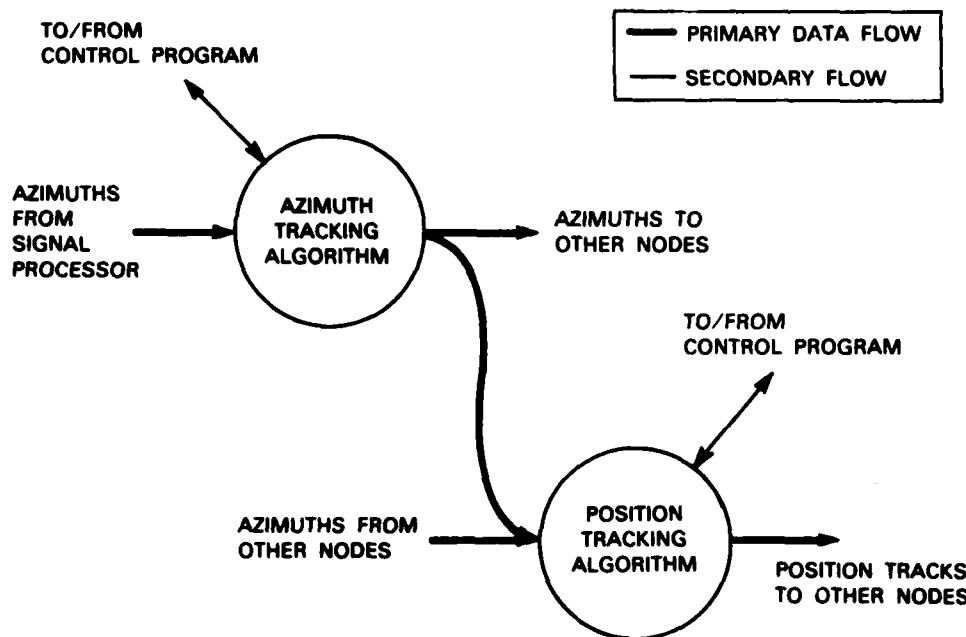


Figure IV-1. Software organization and data flows for initial two-node acoustic tracking algorithms.

move ahead of real time when using simulated data. Again, the net result is the loss of synchronization within the network. This second case would be impossible with real-time acoustic data, but it did constitute a complication for experimenting with simulated data or prerecorded acoustic data being read from tapes. Fortunately, the amount of past data retained in each node allowed the distributed system to operate even when nodes were several seconds out of synchronization, and this allowed us to experiment with the acoustic tracking performance. Experience with the test-bed system confirmed the many advantages that could be obtained from providing synchronized clocks in the nodes for use by the tracking system.

The standard nodal computer (SNC) that performs tracking did not contain floating point hardware. Software floating point was too slow to be used and all calculations were implemented using scaled integer arithmetic. The dynamic range of scaled integer arithmetic resulted in degraded tracking performance due to round-off errors. Attempts to rescale the calculations proved time-consuming and only slightly fruitful. This experience convinced us that some means should be found to provide a hardware floating point arithmetic capability for the SNCs. A survey of board-level floating point products has been completed and we are now procuring a set of boards to integrate into the test bed.

Otherwise, the experiments yielded few surprises. The tracking performance yielded position estimation errors ranging up to 500 m but typically averaging 200-300 m. These figures are for the case where tracking algorithms provide a separate track for each node pair. The real-time recognition and combining of multiple two-node tracks is a difficult problem which

has been circumvented by newly developed tracking algorithms that are discussed below. These new algorithms are not restricted to tracking with node pairs.

B. DISTRIBUTED MULTISITE TRACKING ALGORITHMS

A new distributed tracking algorithm was described in the previous SATS¹ along with the results of initial validation tests with simulated azimuth measurement data. A more detailed technical description of the algorithm and its performance have been written in the form of a paper for the 1984 American Control Conference.²

A new track initiation algorithm was also described in the previous SATS. The tracking and track initiation algorithms have been integrated with each other and tested through simulation during the current reporting period. A number of problems resulting from the interactions between the two algorithms were identified and solved.

The two algorithms obviously interact when the track initiation algorithm provides a new track for the position tracking algorithm. The two algorithms also interact at other times. Within a node, each new local azimuth measurement is used either by the position tracker to update a position track or by the position track initiator to update its data bases even when no new track is initiated. Other interactions occur between the track initiation algorithm in one node and the tracking algorithm in another node when one node has a position track and the second has only an azimuth track for the same target. This can occur, for example, if a message to alert the second node to the existence of the target is not received or if the second node prematurely deletes its position track on the basis of poor accuracy.

Such interactions have required us to extend the algorithms and modify communications policies. The following is an example. The basic communication strategy for track initialization is to broadcast azimuth tracks to support the calculation of new position tracks. Ideally, azimuth tracks are broadcast only for targets for which position tracks have not been established. Thus, when an azimuth track broadcast is received, it could be used immediately to attempt to initiate a new track. This policy was modified to first check the received azimuth against targets. If a plausible association is found, the azimuth is used to update the appropriate position track and the new position track is broadcast. Upon receipt of the broadcast, the node that initially broadcast the azimuth must associate the position track with its azimuth track, discard the azimuth track and establish the position track locally.

A number of such problems were identified and appropriate modifications made to algorithms and communication strategies.

C. TEST-BED IMPLEMENTATION OF NEW DISTRIBUTED ALGORITHMS

The basic test-bed software design has been completed for the new distributed tracking algorithms. This involved considerable restructuring of both tracking and user interface software. There are fundamental differences between the old and new algorithms in terms of the

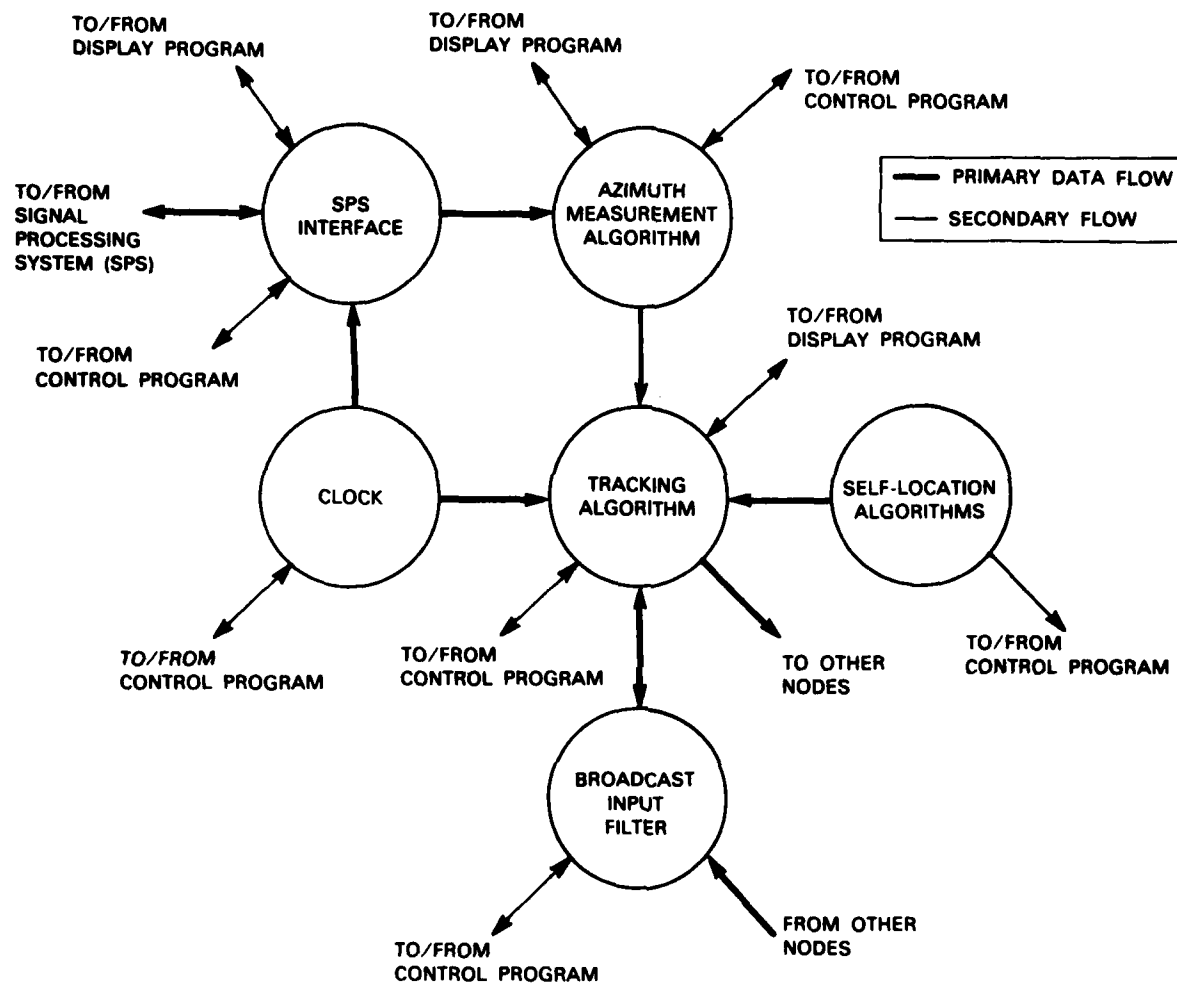


Figure IV-2. Software organization and data flows for new distributed tracking system.

association of position tracks with azimuth tracks, the temporal evolution of the position tracks, and the observability of tracking performance. The new design, that has been developed in response to these differences, represents a more general solution to many DSN problems.

As an example of the differences between the old and new algorithms, consider how surveillance data might be accessed in the DSN. The communication policy for the old algorithms was simple; all azimuth and position track information was broadcast and, therefore, available for analysis and display. The new algorithms can have significantly different private and public data at each node. The new design anticipates the need to access, analyze and display this information for users. It also includes interfaces to a self-location system and provides for test-bed improvements with respect to node synchronization, network control, communications and displays.

Figure IV-2 illustrates the major functional elements and data flow of the new software design. As with Figure IV-1, the heavy lines show primary data flows and the thin lines show secondary but essential flow.

The tracking algorithm element in the figure includes the closely coupled functions of track initiation, azimuth tracking and target tracking. An interface between the tracking algorithm and self-location algorithm has been defined in anticipation of the development and integration of self-location algorithms. Several additional elements have been added when compared with the previous tracking algorithms.

The broadcast input filter, azimuth measurement module and SPS interface serve as data filtering and conditioning modules for the tracking algorithm. The broadcast input filter decides which data should be forwarded to the tracking algorithm. It can also be used to experiment with broadcast connectivity by allowing only messages from certain nodes to pass through the filter. The azimuth measurement module prefilters azimuth measurements for the tracking algorithm on the basis of measurement quality. The SPS interface isolates possible future changes in the signal processing system and synchronizes the appearance of data from the SPS with the nodal clock.

Each node includes a software clock that is synchronized with nearby nodes. The clock is used to determine when the tracking algorithm is falling behind real-time so that the algorithm can selectively discard data to catch up. The signal processing system interface also uses the clock to regulate delivery of messages and prevent the tracking algorithm from advancing too quickly when simulated or tape recorded data are being used for experiments. The clock also has an adjustable 'time dilation' which allows it to run slower than real-time at a controlled rate. This feature supports the execution of scaled slow-time experiments.

Note that secondary data flow includes interactions with display as well as control programs. The display and control functions are implemented by programs that run in a user interface computer (UIC). Previously, direct nodal interactions were carried out with a UIC control program that in turn mediated data flows to display programs. The change is largely due to the difference in observability of the nodal data for the new algorithms; not

all interesting data is routinely available to the control program. It also reflects a change in the basic mode of internodal communications from message switching through the UIC to distributed communications over an Ethernet or radios.

The older tracking system routinely broadcast all tracks and broadcast communications were simulated by means of serial lines and a communications switching program in the UIC. All tracking data in the system passed through the UIC and could be monitored and displayed there without directly interacting with the nodes. In contrast, the new tracking algorithm does not broadcast all track messages. Therefore, separate access is provided for tracking information that is not normally broadcast. Note, however, that even if broadcast track messages provided sufficient performance information, special data flow would still be needed. The reason is that there is no point in the DSN network which receives all broadcasts. Monitoring of a remote node will require additional multihop communication access from the monitoring or display site to the node. In addition, there is important performance information contained in other nontrack data that are never broadcast during normal operation and the new software provides for access to that information.

The software design is highly modular and completely based upon the use of multiple cooperating processes interacting by means of messages. Each functional element is implemented as a single process or a few cooperating processes. Each process has a main input port for receiving messages and each message has a type. Upon receipt of a message, a process tests the message type and, depending upon the type, invokes one of several routines, passing the message as a parameter. This organization provides for simple, easily testable and extensible software since the processing associated with a particular message type is modularized and the addition of a new message type requires only that the case be included in the test and a new processing module be added.

REFERENCES

1. Semiannual Technical Summary, Distributed Sensor Networks Program, Lincoln Laboratory, M.I.T. (30 September 1983).
2. R.R. Tenney and J.R. Delaney, "A Distributed Aeroacoustic Tracking Algorithm," 1984 American Control Conference, San Diego, CA, 8 June 1984.
3. Semiannual Technical Summary, Distributed Sensor Networks Program, Lincoln Laboratory, M.I.T. (31 March 1983), DTIC AD-A133250/1.

V. KNOWLEDGE-BASED DATA INTERPRETATION

A primary DSN function is to provide users with the best possible description of events taking place within the DSN. The descriptions can be at many different levels of abstraction and may be generated by both numeric and symbolic algorithms. We have begun investigating the application of artificial intelligence concepts to provide the best possible high-level interpretations of DSN data. We have formulated an approach which is strongly geared toward practical demonstration through actual system development. We are focusing upon metalevel interpretation systems that collect and interpret tracking data from DSN nodes. In general, the interpretations must be constantly updated as new data becomes available. Updates will typically be based upon new data, saved data histories and saved descriptions that constitute past interpretations.

We view the high-level interpretation process as consisting of two main components: Expectation-Formation and Diagnosis (See Figure V-1). Expectations about the future DSN outputs are formulated on the basis of past interpretations. When the actual outputs become available, we diagnose the discrepancies between the outputs and the corresponding expectations. Four different kinds of diagnoses have been identified. These are (1) incorrect parameter settings in the DSN system, (2) inherent limitations of system capability, (3) system faults such as communication failures, and (4) incorrect expectations. Such diagnoses are used to modify the expectations in order to form the latest data interpretation. Furthermore, the diagnoses can be used to make changes in the processing strategy of the DSN system so that it is better matched to the characteristics of the current scenario.

We decided to first investigate the diagnosis component of data interpretation. This decision was primarily based on the fact that considerable attention has already been given to the diagnosis function in the field of artificial intelligence. Furthermore, expert systems development tools are available that are well matched to diagnosis problems.

During the current reporting period, we constructed an experimental 40-rule DSN diagnosis system that performs simple diagnosis for single aircraft scenarios. The system was constructed using the EMYCIN expert system development tool. Given descriptions of discrepancies between expectations and DSN output, our rule-based diagnosis system provides a plausible cause of the discrepancies. Because of the limited size of the system, it could produce only some of the simpler diagnoses such as 'aircraft was out of acoustic detection range' and 'signal processing module failed'. More sophisticated diagnoses would have required a far greater number of rules.

As a second experiment, we have started building another rule-based diagnosis system using the YAPS tool, which is an extension of the more well-known OPS-5 tool. Compared to EMYCIN, this tool imposes far fewer constraints on the system design. Consequently, we are building a more sophisticated system, which, in particular, will be able to deal with multiple aircraft scenarios. This system is being implemented on a Symbolics 3600 Lisp Machine. We intend to expand this system to a size large enough to reach the limits of a rule-based

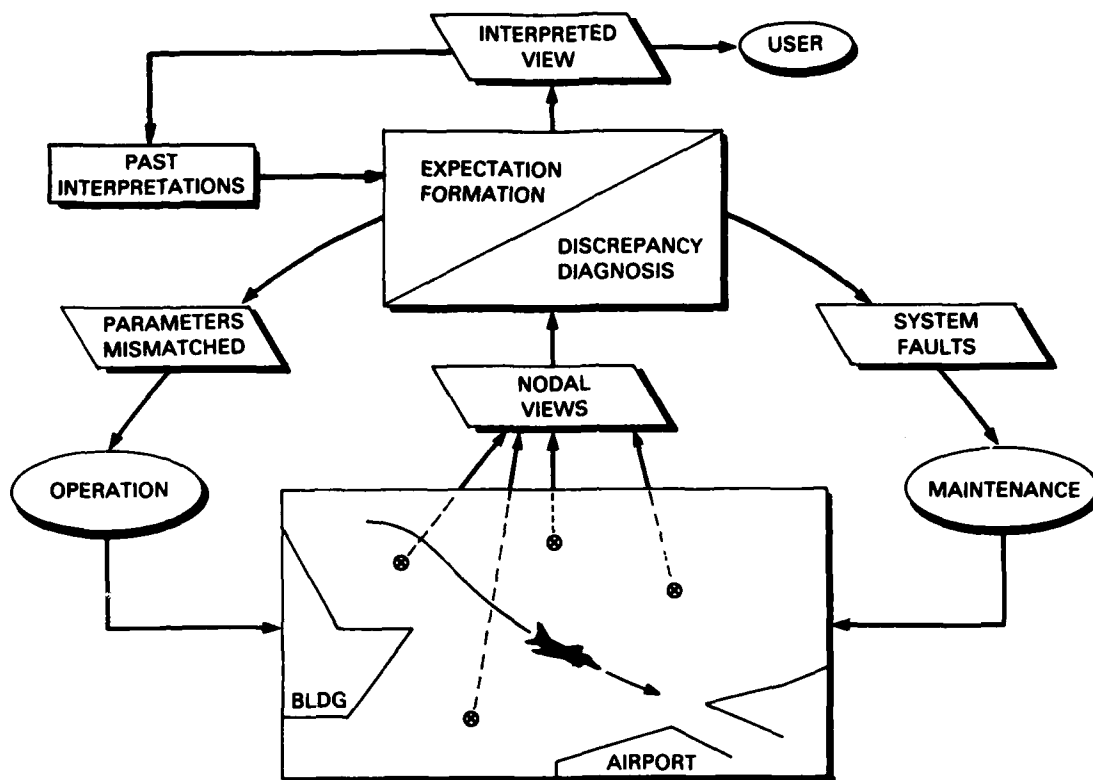


Figure V-1. Organization of knowledge-based DSN data interpretation in terms of the expectation-formation and diagnosis function.

design for the DSN diagnosis problem. One limitation that we anticipate is that causal reasoning from first principles appears to be very useful for sophisticated DSN diagnosis, and rule-based systems are not well matched to implementing this form of reasoning.

Since the beginning of this year, we have also been working in close collaboration with Professor Victor Lesser of the University of Massachusetts and the MIT Artificial Intelligence Laboratory to design and eventually develop a knowledge-based diagnosis system that attacks the DSN diagnosis problem in its entirety. We are imposing no *a priori* restrictions for the system to use a particular expert systems tool or a particular knowledge representation style. The design that is evolving attempts to overcome the difficulties we have encountered in forcing DSN diagnosis expertise into standard expert system architectures such as YAPS or EMYCIN. The new design provides multiple reasoning methods and the use of multiple levels of abstraction to perform DSN diagnosis.

VI. COMMUNICATIONS

We have been engaged in two primary communication related activities during this reporting period. First, we have continued with the development of a radio communication capability for the DSN test bed based upon the radios that are being developed under the Communication Networks Technology program. Second, we have started to add an Ethernet local area network that will provide substantial improvements in communication capabilities and experimental flexibility.

A. RADIO COMMUNICATION

The development of a radio communications capability has proceeded along several lines. The detailed software design for the higher levels of our broadcast protocol was completed. Major elements have now been coded and we have started debugging in a UNIX test environment. A two-node software and hardware test set-up was designed and constructed. The set-up includes back-to-back radio unit interface (RUI) boards which form the bridge between standard nodal computers and the Communications Technology Radios. The test set-up will provide an environment for debugging software drivers for the interface board and for integrating the higher level protocols with those drivers. The following provides more details concerning the software design, implementation status and the two-node test set-up.

DSN radio protocols must ultimately provide for many services including data-link level local broadcast, point-to-point service and a ranging protocol to support self-location. We have concentrated upon the design and development of a simple local broadcast protocol for the test-bed because that service is essential for DSN experimentation and because it can eventually serve as the basis for other services. The local broadcast service is being implemented as cooperating processes that operate within a single processor in the standard test bed nodal computers. The software can eventually be extended to provide other services by the addition of more processes.

Our process organization to implement the broadcast protocol is illustrated in Figure VI-1. The primary means of interprocess communication is through queues. This permits processes to operate concurrently and independently as long as their input queues are not empty and is intended to make efficient use of computational resources. A brief description of the functions of each of the processes follows.

The receive and retrieve processes combine to transfer received messages from the radio to the application program interface. The receive portion interacts with lower level radio software. It is an interrupt service routine that responds to interrupts posted by the 8089 input-output processor on the RUI board when messages are received by the radio.

The remaining select, scheduler, transmit, status and collect processes all have to do with sending commands to the radio.

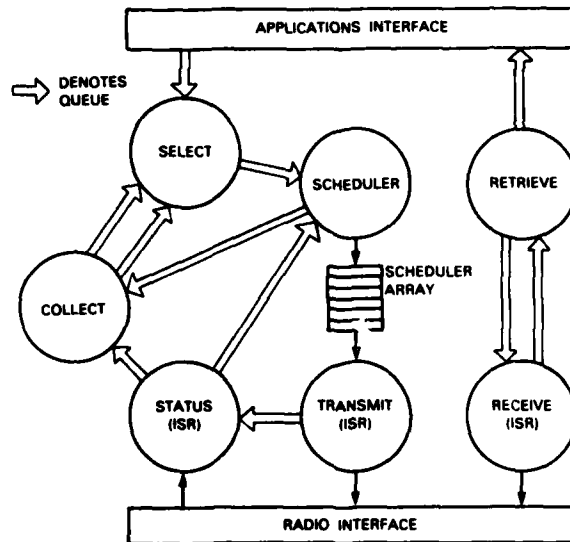


Figure VI-1. Organization of software to implement DSN radio broadcast protocol.

The select process collects messages from the applications interface and assembles them into the appropriate data structures for transmission. The assembled data structures are then passed to the scheduler which schedules commands for transmission. The structures written by the scheduler represent time slots and determine the times at which commands will be sent to the radio.

The transmit process sends commands to the radio at the times established by the scheduler. If there is no command scheduled for a particular time slot, a default is used that causes the radio to remain in receive mode. The transmit process is time-critical in that it must respond to interrupts from the radio with a command every 10 ms. Since the transmit and scheduler processes share data, their interaction is also critical and considerable care has been taken in the detailed design to assure that they cannot interfere with each other.

When the transmit process is finished with a command, it places it on a queue for the status process. The status process reads a status packet returned from the radio during each 10 ms slot and matches the status to commands in the return queue. If the status packet indicates that the command has been successfully transmitted, then the status process sends the structure to the collect process for disassembly and return to the pools of available data structures. Otherwise, if transmission was not successful, the status process returns the command to the scheduler for scheduling and retransmission.

The broadcast protocol involves several hundred lines of code. Since nodal computers provide limited debugging tools, we are doing as much development, testing and debugging as possible in the UNIX software environment. Test procedures and a library of UNIX routines have been developed that provide an environment very similar to the nodal computer

operating system. This supports simulated operation of the software almost exactly as in the nodal computers but not in real-time. At the present time, the scheduler, select, transmit and status processes have been written and are operational in this non-real-time environment. Once all elements are operational in this environment, we will proceed to the real-time environment provided by the two-node test set-up described below.

During the current reporting period, we have assembled a hardware configuration to allow development and testing of the communication protocol software in a realistic environment, but without radios. The test configuration involves two RUI boards wired back-to-back as shown in Figure VI-2. The FIFOs on the board are connected so that the command FIFO on one sends data to the status FIFO of the other and the transmit FIFO sends data to the receive and observation FIFOs of the other. Signals for framing, transmit abort, and radio reset are provided as outputs of registers in the control logic of the RUI. The radio 10 ms clock 'tick' is simulated by an external timer circuit and is presented simultaneously to both boards. The back-to-back RUIs are installed in two standard nodal computers located in a single rack. Software can be loaded into the computers and tested. The test set-up supports testing of one-way transmission in which one node transmits while the other receives and two-way transmission in which both nodes transmit and receive.

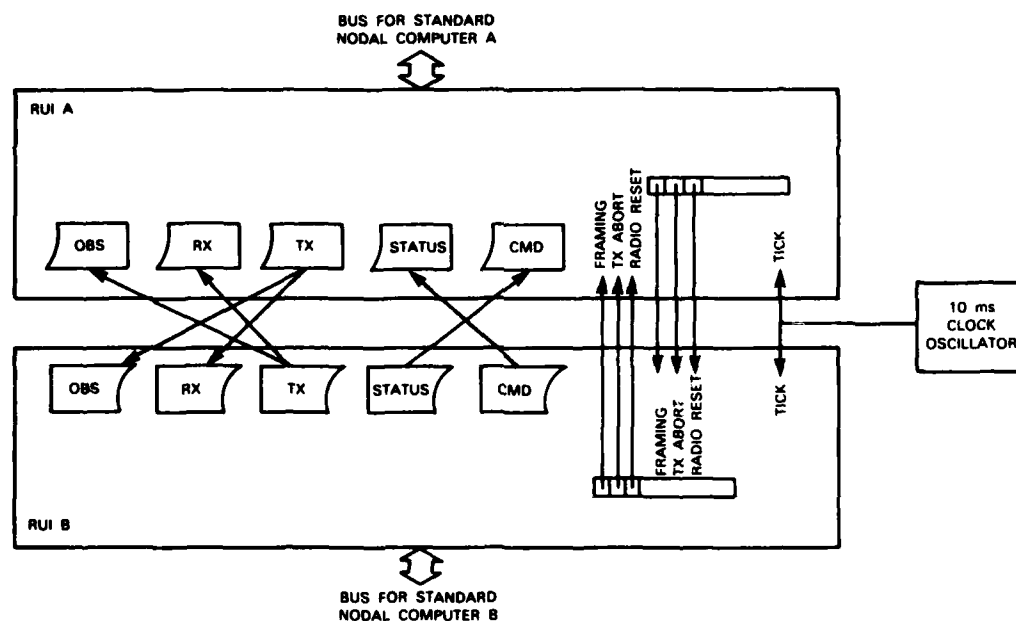


Figure VI-2. Two-node radio communication hardware/software test set-up.

Test software for the RUI board hardware has been developed for the test set-up and is being used for final debugging of the RUI boards. The RUI hardware operates correctly in the one-way test mode but operates only intermittently in the two-way test mode. This appears to be due to undocumented features of the input-output processor chip on the RUI board, and we are now modifying the program for that chip to solve the problem.

B. LOCAL AREA NETWORK

We have begun the integration of a local area network capability into the DSN test bed. After a review of available options, we have decided to employ an Ethernet system for this purpose.

The objective is to enhance the flexibility and utility of the test bed in a number of ways. First, the Ethernet represents a general way in which we can interconnect subsystems within individual nodes. For example, the TV subsystem will be connected to the standard nodal computer by Ethernet and other sensory subsystems could be easily added. Second, the local net will provide for a high speed internodal communication alternative to radios for many experiments that can be conducted in a laboratory environment where nodes are not physically separated from each other by large distances. Third, the Ethernet provides a convenient way to interface users to nodes and to the test bed. For example, we have several computers, including a LISP machine, that can provide user access and system control and can easily be interfaced with any node through Ethernet technology. Finally, a major short term goal is to provide improved internodal broadcast capability on an interim basis while the radio system is being developed for the test bed. The Ethernet will provide a datagram service to application programs similar to the service that is currently provided through serial lines and a central switch. The increased communication capability will support many experiments that heretofore were not possible.

Present plans call for the development of a point-to-point and broadcast datagram service for the test bed. With this, we can replace the serial lines and message switch that are being used to simulate the broadcast medium. The new tracking software described in Section IV-C is being developed to interface directly with the Ethernet datagram service. The user interface program that now operates on our PDP-11/70 Unix system will be restructured and interfaced to the Ethernet as part of the development of the new tracking software and integration of the Ethernet into the test bed. Subsequently, we will add additional Ethernet communication functions as specific requirements are identified.

The integration of the Ethernet requires driver software at the ISO/OSI model data link level of protocol. We have obtained an Ethernet front-end processor board for evaluation and software development and have installed it in one of our standard nodal computers. A link level device driver is now being developed for the board. We have also obtained an Ethernet interface board for our PDP-11/70 computer and have procured Ethernet software for installation on that system.

VII. SELF-LOCATION

We have investigated the problem of DSN network self-location as a distributed estimation problem and have derived new distributed estimation-theoretic algorithms for determination of the network geometry. Our goal is to develop decentralized self-location algorithms that use range measurements between nodes as the raw data and optimally treat range measurement errors. Our algorithms require initial estimates of nodal positions. A longer range goal is to merge our algorithms with those being developed at Columbia University that do not require initial estimates but do not optimally treat measurement errors.

The major assumptions we have made in pursuing this research are the following. First, nodes are stationary and lie in a two-dimensional plane. The case of nodes in three dimensions or on a known two-dimensional nonplanar surface could be treated but would introduce unnecessary complexity for initial development and testing of algorithms. Second, there are no multipath errors that introduce consistent biases in range measurements. Third, reasonably good initial estimates are available for node locations. Fourth, there is sufficient connectivity in the network to allow a solution.

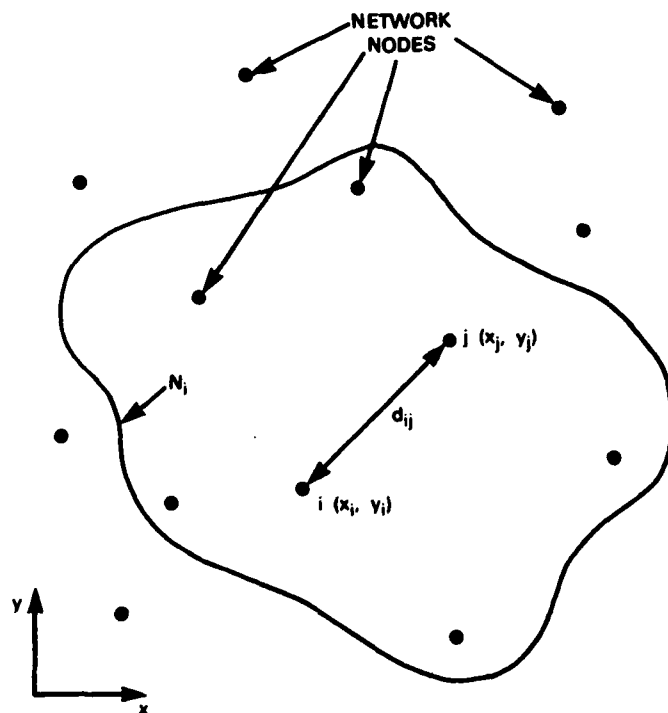
As noted above, the self-location problem can be formulated as a statistical estimation problem. In particular, we have assumed white Gaussian noise on range measurements, and used a Bayesian approach to obtain the probability distribution of the coordinates of each node. We then develop algorithms to find the (global) maximum of this function, and use the maximizing value as our estimate.

Figure VII-1 shows a possible configuration for a portion of a large network, where each dot is a node and the x^i and y^i are the position coordinate parameters to be estimated for each node i . The contour line N^i encloses a set of neighbor nodes for node i , i.e., those nodes which can communicate directly with node i . This contour is determined by the type of radio at the node, the electromagnetic environment and the local geography and is not known *a priori*. The measurements between nodes are the distances d^i , which are computed using time-of-flight data.

The self-location problem is complicated because it is a nonlinear estimation problem and we want to develop a distributed solution (i.e., an algorithm that can be executed independently in each node without centralized control). In order to attack and understand the different aspects of the problem in an organized manner, we have investigated four different versions of the self-location problem. These are:

- (1) Centralized solution of a self-location problem with linear equations.
- (2) Distributed solution of a problem with linear equations.
- (3) Centralized solution of a self-location problem with nonlinear equations.
- (4) Distributed solution of a problem with nonlinear equations.

The fourth one of these problems is the actual DSN self-location problem.



143118-N

Figure VII-1. Typical two-dimensional network layout with contour N: showing extent of communication region for node i.

For the first case, we assumed that the measurements are estimates of the position difference between nodes. The centralized problem is then a linear estimation problem that can be solved using Kalman filter algorithms. For the second problem, we assume the same linear measurements but use them in a decentralized computation. We have derived a necessary condition that each node can maximize a probability density and thereby estimate its own position. The resulting estimate involves the estimates from other nodes in the network. A convenient property of the equations is that a node need know only the position estimates of those nodes with which it can directly communicate. This is true as long the node can communicate with every node for which there is a position estimate relative to itself. Initial work with the linearized problem provided the insight needed to develop the solution to the more important and difficult nonlinear problem that results when range, not position differences, are available as measurements.

For mathematical convenience, we have chosen to use the square of range measurements as the observations for the nonlinear self-location problem. This yields fourth order equations in the positions of the nodes. We have derived both centralized and decentralized algorithms to solve these equations. The centralized solution is based on Newton's method, and involves an iteration about an initial estimate. The decentralized solution results in a set of

simultaneous cubic equations at each node. We are now investigating alternatives to efficiently obtain solutions to these equations for the distributed case. As in the linear case, nodes require access only to information from neighbors.

For the decentralized algorithms, a communications strategy has been formulated to determine when a node should update and broadcast its current position estimate. Recall that the estimate at each node depends on the estimates at the other nodes. A change that one node makes will affect other estimates, as least as least as long as the solutions have not converged. A simulation of the distributed self-location process, including the communication policy, is now being implemented to investigate this important issue of convergence.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DD FORM 1473 EDITION OF 1 NOV 66 IS OBSOLETE
1 Jan 73

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)